

Nama : Daffa Aryaputra
Nim : 2311102272
Kelas : 11IF06

Modul 15

Soal 1

Code :

```
package main
```

```
import "fmt"
```

```
type set [2022]int
```

```
// Function to check if a value exists in a set
func exist(T set, n int, val int) bool {
    for i := 0; i < n; i++ {
        if T[i] == val {
            return true
        }
    }
    return false
}
```

```
// Function to input values into a set
func inputSet(T *set, n *int) {
    var x int
    for {
        fmt.Scan(&x)
        if exist(*T, *n, x) {
            continue // Skip duplicate values
        }
        if x == -1 { // -1 indicates the end of input
            break
        }
        T[*n] = x
        *n++
    }
}
```

```
// Function to find the intersection of two sets
func findIntersection(T1 set, n1 int, T2 set, n2 int, T3 *set, n3 *int)
{
    for i := 0; i < n1; i++ {
        if exist(T2, n2, T1[i]) {
            (*T3)[*n3] = T1[i]
            *n3++
        }
    }
}
```

```
// Function to print a set
func printSet(T set, n int) {
    for i := 0; i < n; i++ {
        if i > 0 {
            fmt.Print(" ")
        }
        fmt.Print(T[i])
    }
    fmt.Println()
}
```

```
func main() {
    var s1_2311102272, s2_Daffa, s3_IF set
    var n1, n2, n3 int
```

```
    fmt.Println("Input set 1 (end with -1):")
    inputSet(&s1_2311102272, &n1)
```

```
    fmt.Println("Input set 2 (end with -1):")
    inputSet(&s2_Daffa, &n2)
```

```
    findIntersection(s1_2311102272, n1, s2_Daffa, n2, &s3_IF, &n3)
```

```
    fmt.Println("Intersection of sets:")
    printSet(s3_IF, n3)
}
```

```

modul15.go 1 X modul15_2.go 1 modul15_3.go 1 modul15_4.go 2 modul15_5.go 1
E:\> golang semester 3 > -> modul15.go > main
1 package main
2
3 import "fmt"
4
5 type set [2022]int
6
7 // Function to check if a value exists in a set
8 func exist(T set, n int, val int) bool {
9     for i := 0; i < n; i++ {
10         if T[i] == val {
11             return true
12         }
13     }
14     return false
15 }
16
17 // Function to input values into a set
18 func inputSet(T *set, n *int) {
19     var x int
20     for {
21         fmt.Scan(&x)
22         if exist(*T, *n, x) {
23             continue // Skip duplicate values
24         }
25         if x == -1 { // -1 indicates the end of input
26             break
27         }
28         T[*n] = x
29         *n++
30     }
31 }
32
33 // Function to find the intersection of two sets
34 func findIntersection(T1 set, n1 int, T2 set, n2 int, T3 *set, n3 *int) {
35     for i := 0; i < n1; i++ {
36         if exist(T2, n2, T1[i]) {
37             (*T3)[*n3] = T1[i]
38             *n3++
39         }
40     }
41 }
42
43 // Function to print a set
44 func printSet(T set, n int) {
45     for i := 0; i < n; i++ {
46         if i > 0 {
47             fmt.Print(" ")
48         }
49         fmt.Print(T[i])
50     }
51     fmt.Println()
52 }
53
54 func main() {
55     var s1 2311182272, s2 Daffa, s3 IF set
56     var n1, n2, n3 int
57
58     fmt.Println("Input set 1 (end with -1):")
59     inputSet(&s1, 2311182272, &n1)
60
61     fmt.Println("Input set 2 (end with -1):")
62     inputSet(&s2, Daffa, &n2)
63
64     findIntersection(s1, 2311182272, n1, s2, Daffa, n2, &s3, &n3)
65
66     fmt.Println("Intersection of sets:")
67     printSet[s3, n3]
68 }

```

```

PS C:\Users\ACER> go run "e:\golang semester 3\modul15.go"
Input set 1 (end with -1):
1 1 -1
Input set 2 (end with -1):
1 1 -1
Intersection of sets:
1
PS C:\Users\ACER>

```

Soal 2

Code :

```
package main
```

```
import (  
    "fmt"  
    "strings"  
)
```

```
// Konstanta untuk batas maksimum mahasiswa  
const NMax = 51
```

```
// Struct untuk menyimpan data mahasiswa  
type Mahasiswa struct {  
    NIM    string  
    Nama   string  
    Nilai  int  
}
```

```
// Array untuk menyimpan data mahasiswa  
type ArrayMahasiswa [NMax]Mahasiswa
```

```
func main() {  
    var mahasiswaList_2311102272 ArrayMahasiswa  
    var n int
```

```
    // Memasukkan jumlah data mahasiswa  
    fmt.Print("Masukkan jumlah mahasiswa (max ", NMax, "): ")  
    fmt.Scan(&n)
```

```
    if n > NMax || n < 1 {  
        fmt.Println("Jumlah mahasiswa tidak valid!")  
        return  
    }
```

```
    // Memasukkan data mahasiswa  
    for i := 0; i < n; i++ {  
        fmt.Println("Masukkan data mahasiswa ke-", i+1)  
        fmt.Print("NIM: ")  
        fmt.Scan(&mahasiswaList_2311102272[i].NIM)  
        fmt.Print("Nama: ")  
        fmt.Scan(&mahasiswaList_2311102272[i].Nama)  
        fmt.Print("Nilai: ")  
        fmt.Scan(&mahasiswaList_2311102272[i].Nilai)  
    }
```

```
    // Menampilkan semua data mahasiswa  
    fmt.Println("\nData Mahasiswa:")  
    for i := 0; i < n; i++ {
```

```

        fmt.Printf("NIM: %s, Nama: %s, Nilai: %d\n",
mahasiswaList_2311102272[i].NIM, mahasiswaList_2311102272[i].Nama,
mahasiswaList_2311102272[i].Nilai)
    }

```

```

// Mencari mahasiswa berdasarkan NIM
fmt.Print("\nMasukkan NIM untuk pencarian: ")
var cariNIM string
fmt.Scan(&cariNIM)

```

```

index := cariMahasiswa(mahasiswaList_2311102272, n, cariNIM)
if index == -1 {
    fmt.Println("Mahasiswa dengan NIM tersebut tidak ditemukan.")
} else {
    fmt.Printf("Mahasiswa ditemukan: NIM: %s, Nama: %s,
Nilai: %d\n",
        mahasiswaList_2311102272[index].NIM,
mahasiswaList_2311102272[index].Nama,
mahasiswaList_2311102272[index].Nilai)

```

```

// Mencari nilai terbesar dengan NIM tertentu
maxNilai := cariNilaiTertinggi(mahasiswaList_2311102272, n,
cariNIM)
fmt.Printf("Nilai terbesar dari NIM %s adalah %d\n", cariNIM,
maxNilai)
}

```

```

// Fungsi untuk mencari mahasiswa berdasarkan NIM
func cariMahasiswa(arr ArrayMahasiswa, n int, nim string) int {
    for i := 0; i < n; i++ {
        if strings.EqualFold(arr[i].NIM, nim) {
            return i
        }
    }
    return -1
}

```

```

// Fungsi untuk mencari nilai terbesar berdasarkan NIM
func cariNilaiTertinggi(arr ArrayMahasiswa, n int, nim string) int {
    maxNilai := -1
    for i := 0; i < n; i++ {
        if strings.EqualFold(arr[i].NIM, nim) {
            if arr[i].Nilai > maxNilai {
                maxNilai = arr[i].Nilai
            }
        }
    }
    return maxNilai
}

```

}

```

1 // github.com/0x00sec/0x00sec/2.0.0/0x00sec/2.0.0/0x00sec
2 package main
3
4 import {
5     "fmt"
6     "strings"
7 }
8
9 // Struktur untuk data serangan
10 const WPA = 1
11
12 // Struktur untuk serangan data serangan
13 type Serangan struct {
14     ID string
15     Nama string
16     Nilai int
17 }
18
19 // Array untuk menyimpan data serangan
20 type ArraySerangan []Serangan
21
22 func main() {
23     var SeranganList []Serangan
24     var s Ser
25
26     // Menampilkan judul data serangan
27     fmt.Println("Menukar Judul Serangan (Nama, WPA, ID):")
28     fmt.Scan(&s)
29
30     if s.WPA == 0 {
31         fmt.Println("Judul serangan tidak valid")
32         return
33     }
34
35     // Menampilkan data serangan
36     for i := 0; i < s.ID; i++ {
37         fmt.Println("Masukkan data serangan ke-", i)
38         fmt.Scan(&s)
39         SeranganList[i].ID = s.ID
40         SeranganList[i].Nama = s.Nama
41         SeranganList[i].Nilai = s.Nilai
42     }
43
44     // Menampilkan semua data serangan
45     fmt.Println("Data Serangan:")
46     for i := 0; i < s.ID; i++ {
47         fmt.Println("ID: %d, Nama: %s, Nilai: %d", SeranganList[i].ID, SeranganList[i].Nama, SeranganList[i].Nilai)
48     }
49
50     // Menampilkan serangan berdasarkan ID
51     fmt.Println("Serangan ID yang dicari:")
52     var cariID string
53     fmt.Scan(&cariID)
54
55     index := cariID(SeranganList[i].ID, s, cariID)
56     if index == -1 {
57         fmt.Println("Serangan dengan ID tersebut tidak ditemukan.")
58     } else {
59         fmt.Println("Serangan ditemukan: ID: %d, Nama: %s, Nilai: %d",
60             SeranganList[index].ID, SeranganList[index].Nama, SeranganList[index].Nilai)
61     }
62
63     // Menampilkan serangan dengan ID tertentu
64     cariID := cariID(SeranganList[i].ID, s, cariID)
65     fmt.Println("Nilai serangan dari ID: %d, cariID, cariID, cariID")
66 }
67
68 // Fungsi untuk mencari serangan berdasarkan ID
69 func cariID(SeranganList []Serangan, s int, cariID string) int {
70     for i := 0; i < s; i++ {
71         if strings.EqualFold(s[i].ID, cariID) {
72             return i
73         }
74     }
75     return -1
76 }
77
78 // Fungsi untuk mencari nilai serangan berdasarkan ID
79 func cariNilai(SeranganList []Serangan, s int, cariID string) int {
80     cariID := -1
81     for i := 0; i < s; i++ {
82         if strings.EqualFold(s[i].ID, cariID) {
83             cariNilai = s[i].Nilai
84         }
85     }
86     return cariNilai
87 }
88
89 }

```

```
PS C:\Users\ACER> go run "e:\golang semester 3\modul15_2.go"
Masukkan jumlah mahasiswa (max 51): 2
Masukkan data mahasiswa ke- 1
NIM: 2331110222
Nama: ucup
Nilai: 80
Masukkan data mahasiswa ke- 2
NIM: 2311102272
Nama: daffa
Nilai: 90

Data Mahasiswa:
NIM: 2331110222, Nama: ucup, Nilai: 80
NIM: 2311102272, Nama: daffa, Nilai: 90

Masukkan NIM untuk pencarian: 2311102272
Mahasiswa ditemukan: NIM: 2311102272, Nama: daffa, Nilai: 90
Nilai terbesar dari NIM 2311102272 adalah 90
PS C:\Users\ACER> 
```

Soal 3

Code :

```
package main
```

```
import (  
    "fmt"  
    "math"  
)
```

```
const nProv = 34 // Jumlah provinsi yang ditangani program
```

```
// Struct untuk menyimpan data tentang provinsi  
type Provinsi struct {  
    nama      string // Nama provinsi  
    populasi  int     // Populasi provinsi  
    tumbuh   float64 // Tingkat pertumbuhan populasi (dalam format  
desimal, misalnya 0.03 untuk 3%)  
}
```

```
func main() {  
    var prov [nProv]Provinsi // Array untuk menyimpan data semua  
provinsi  
    var namaDicari string    // Nama provinsi yang akan dicari
```

```
    // Input data provinsi  
    for i := 0; i < nProv; i++ {  
        fmt.Printf("Masukkan data provinsi ke-%d (nama populasi  
tingkat_pertumbuhan): ", i+1)  
        fmt.Scan(&prov[i].nama, &prov[i].populasi, &prov[i].tumbuh)  
    }  
    fmt.Print("Masukkan nama provinsi yang ingin dicari: ")  
    fmt.Scan(&namaDicari) // Membaca nama provinsi yang dicari
```

```
    // Menampilkan nama provinsi dengan angka pertumbuhan tercepat  
    indeksTercepat := provinsiTercepat_2311102272(prov)  
    fmt.Printf("Provinsi dengan pertumbuhan tercepat: %s\n",  
prov[indeksTercepat].nama)
```

```
    // Menampilkan indeks provinsi yang dicari  
    indeksDicari := indeksProvinsi(prov, namaDicari)  
    if indeksDicari != -1 {  
        fmt.Printf("Indeks provinsi %s: %d\n", namaDicari, indeksDicari)  
    } else {  
        fmt.Printf("Provinsi %s tidak ditemukan.\n", namaDicari)  
    }  
}
```

```
    // Menampilkan prediksi jumlah penduduk untuk provinsi dengan  
pertumbuhan di atas 2%
```



```

    fmt.Println("Prediksi populasi untuk provinsi dengan pertumbuhan di
atas 2%:")
    prediksi(prov)
}

```

```

// Fungsi untuk menemukan indeks provinsi dengan tingkat pertumbuhan
tercepat
func provinsiTercepat_2311102272(prov [nProv]Provinsi) int {
    maxIndex := 0 // Asumsikan provinsi pertama memiliki pertumbuhan
    tercepat
    for i := 1; i < nProv; i++ {
        if prov[i].tumbuh > prov[maxIndex].tumbuh {
            maxIndex = i
        }
    }
    return maxIndex
}

```

```

// Fungsi untuk mencari indeks provinsi berdasarkan nama
func indeksProvinsi(prov [nProv]Provinsi, nama string) int {
    for i := 0; i < nProv; i++ {
        if prov[i].nama == nama {
            return i // Mengembalikan indeks jika ditemukan
        }
    }
    return -1 // Jika tidak ditemukan
}

```

```

// Fungsi untuk menampilkan prediksi populasi untuk provinsi dengan
tingkat pertumbuhan di atas 2%
func prediksi(prov [nProv]Provinsi) {
    for i := 0; i < nProv; i++ {
        if prov[i].tumbuh > 0.02 { // Cek jika tingkat pertumbuhan
lebih dari 2%
            prediksiPopulasi := int(math.Round(float64(prov[i].populasi)
* (1 + prov[i].tumbuh)))
            fmt.Printf("%s: %d\n", prov[i].nama, prediksiPopulasi)
        }
    }
}

```

```

1 // modul15.go
2
3 package main
4
5 import (
6     "fmt"
7     "math"
8 )
9
10 const nProv = 34 // Jumlah provinsi yang ditangani program
11
12 // Struct untuk menyimpan data tentang provinsi
13 type Provinsi struct {
14     nama      string // Nama provinsi
15     populasi int // Populasi provinsi
16     tumbuh float64 // Tingkat pertumbuhan populasi (dalam format desimal, misalnya 0.03 untuk 3%)
17 }
18
19 func main() {
20     var prov []Provinsi // Array untuk menyimpan data semua provinsi
21     var namaDicari string // Nama provinsi yang akan dicari
22
23     // Input data provinsi
24     for i := 0; i < nProv; i++ {
25         fmt.Printf("Masukkan data provinsi ke-%d (nama populasi tingkat_pertumbuhan): ", i+1)
26         fmt.Scan(&prov[i].nama, &prov[i].populasi, &prov[i].tumbuh)
27     }
28     fmt.Print("Masukkan nama provinsi yang ingin dicari: ")
29     fmt.Scan(&namaDicari) // Membaca nama provinsi yang dicari
30
31     // Menampilkan nama provinsi dengan angka pertumbuhan tercapat
32     indeksTercapat := provinsiTercapat_2311182272(prov)
33     fmt.Printf("Provinsi dengan pertumbuhan tercapat: %s\n", prov[indeksTercapat].nama)
34
35     // Menampilkan indeks provinsi yang dicari
36     indeksDicari := indeksProvinsi(prov, namaDicari)
37     if indeksDicari != -1 {
38         fmt.Printf("Indeks provinsi %s: %d\n", namaDicari, indeksDicari)
39     } else {
40         fmt.Printf("Provinsi %s tidak ditemukan.\n", namaDicari)
41     }
42
43     // Menampilkan prediksi jumlah penduduk untuk provinsi dengan pertumbuhan di atas 2%
44     fmt.Println("Prediksi populasi untuk provinsi dengan pertumbuhan di atas 2%:")
45     prediksi(prov)
46 }
47
48 // Fungsi untuk menemukan indeks provinsi dengan tingkat pertumbuhan tercapat
49 func provinsiTercapat_2311182272(prov []Provinsi) int {
50     maxIndex := 0 // Asumsikan provinsi pertama memiliki pertumbuhan tercapat
51     for i := 1; i < nProv; i++ {
52         if prov[i].tumbuh > prov[maxIndex].tumbuh {
53             maxIndex = i
54         }
55     }
56     return maxIndex
57 }
58
59 // Fungsi untuk mencari indeks provinsi berdasarkan nama
60 func indeksProvinsi(prov []Provinsi, nama string) int {
61     for i := 0; i < nProv; i++ {
62         if prov[i].nama == nama {
63             return i // Pengembalian indeks jika ditemukan
64         }
65     }
66     return -1 // Jika tidak ditemukan
67 }
68
69 // Fungsi untuk menampilkan prediksi populasi untuk provinsi dengan tingkat pertumbuhan di atas 2%
70 func prediksi(prov []Provinsi) {
71     for i := 0; i < nProv; i++ {
72         if prov[i].tumbuh > 0.02 { // Cek jika tingkat pertumbuhan lebih dari 2%
73             prediksiPopulasi := int(math.Round(float64(prov[i].populasi) * (1 + prov[i].tumbuh)))
74             fmt.Printf("%s: %d\n", prov[i].nama, prediksiPopulasi)
75         }
76     }
77 }

```

PROBLEMS 131 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ACER> go run "e:\golang semester 3\modul15_3.go"

Masukkan data provinsi ke-1 (nama populasi tingkat_pertumbuhan): Jakarta 10000 0,03

Masukkan data provinsi ke-2 (nama populasi tingkat_pertumbuhan): Bandung 20000 0,02

Soal 4

Code :

```
package main
```

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

```
func main() {  
    var input_2311102272 int  
    var data []int
```

```
    fmt.Println("Masukkan bilangan (akhiri dengan -5313541):")  
    for {  
        fmt.Scan(&input_2311102272)
```

```
        if input_2311102272 == -5313541 {  
            break  
        }  
    }
```

```
    if input_2311102272 == 0 {  
        if len(data) > 0 {  
            sort.Ints(data)  
            median := calculateMedian(data)  
            fmt.Printf("Median: %d\n", median)  
        }  
    } else {  
        data = append(data, input_2311102272)  
    }  
}
```

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

```

: > golang semester 3 > -go modul15_4.go > main
1  package main
2
3  import (
4      "fmt"
5      "sort"
6  )
7
8  func main() {
9      var input_2311102272 int
10     var data []int
11
12     fmt.Println("Masukkan bilangan (akhiri dengan -5313541):")
13     for {
14         fmt.Scan(&input_2311102272)
15
16         if input_2311102272 == -5313541 {
17             break
18         }
19
20         if input_2311102272 == 0 {
21             if len(data) > 0 {
22                 sort.Ints(data)
23                 median := calculateMedian(data)
24                 fmt.Printf("Median: %d\n", median)
25             }
26         } else {
27             data = append(data, input_2311102272)
28         }
29     }
30 }
31
32 func calculateMedian(data []int) int {
33     n := len(data)
34     if n%2 == 1 {
35         return data[n/2]
36     }
37     return (data[(n/2)-1] + data[n/2]) / 2
38 }
39

```

```

PS C:\Users\ACER> go run "e:\golang semester 3\modul15_4.go"
Masukkan bilangan (akhiri dengan -5313541):
7 23 11 0 5 19 2 29 3 13 17 0 -5313541
Median: 11
Median: 12
PS C:\Users\ACER>

```

Soal 5

Code :

```
package main
```

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

```
const NMAX = 1000000
```

```
// Struct untuk menyimpan data partai  
type Partai struct {  
    ID    int  
    Suara int  
}
```

```
// Tipe untuk array partai  
type TabPartai []Partai
```

```
func main() {  
    var tabPartai_2311102272 TabPartai  
    var suara int  
    partaiMap := make(map[int]int)
```

```
    fmt.Println("Masukkan perolehan suara untuk setiap partai (akhiri  
dengan -1):")
```

```
    // Input perolehan suara  
    for {  
        fmt.Scan(&suara)  
        if suara == -1 {  
            break  
        }  
        partaiMap[suara]++  
    }
```

```
    // Memindahkan data dari map ke slice  
    for id, totalSuara := range partaiMap {  
        tabPartai_2311102272 = append(tabPartai_2311102272, Partai{ID:  
id, Suara: totalSuara})  
    }
```

```
    // Mengurutkan secara descending berdasarkan jumlah suara  
    sort.Slice(tabPartai_2311102272, func(i, j int) bool {  
        if tabPartai_2311102272[i].Suara ==  
tabPartai_2311102272[j].Suara {
```

```

        return tabPartai_2311102272[i].ID <
tabPartai_2311102272[j].ID // Jika suara sama, urutkan berdasarkan ID
partai
    }
    return tabPartai_2311102272[i].Suara >
tabPartai_2311102272[j].Suara
    })

```

```

// Menampilkan hasil
fmt.Println("\nHasil perolehan suara:")
for _, partai := range tabPartai_2311102272 {
    fmt.Printf("(%d:%d) ", partai.ID, partai.Suara)
}
fmt.Println()
}

```

```

E > golang semester3 > modu15_5.go > main
1  package main
2
3  import (
4      "fmt"
5      "sort"
6  )
7
8  const NMAX = 1000000
9
10 // Struct untuk menyimpan data partai
11 type Partai struct {
12     ID    int
13     Suara int
14 }
15
16 // Tipe untuk array partai
17 type TabPartai []Partai
18
19 func main() {
20     var tabPartai_2311102272 TabPartai
21     var suara int
22     partaiMap := make(map[int]int)
23
24     fmt.Println("Masukkan perolehan suara untuk setiap partai (akhiri dengan -1):")
25
26     // Input perolehan suara
27     for {
28         fmt.Scan(&suara)
29         if suara == -1 {
30             break
31         }
32         partaiMap[suara]++
33     }
34
35     // Memindahkan data dari map ke slice
36     for id, totalSuara := range partaiMap {
37         tabPartai_2311102272 = append(tabPartai_2311102272, Partai{ID: id, Suara: totalSuara})
38     }
39
40     // Mengurutkan secara descending berdasarkan jumlah suara
41     sort.Slice(tabPartai_2311102272, func(i, j int) bool {
42         if tabPartai_2311102272[i].Suara == tabPartai_2311102272[j].Suara {
43             return tabPartai_2311102272[i].ID < tabPartai_2311102272[j].ID // Jika suara sama, urutkan berdasarkan ID partai
44         }
45         return tabPartai_2311102272[i].Suara > tabPartai_2311102272[j].Suara
46     })
47
48     // Menampilkan hasil
49     fmt.Println("\nHasil perolehan suara:")
50     for _, partai := range tabPartai_2311102272 {
51         fmt.Printf("(%d:%d) ", partai.ID, partai.Suara)
52     }
53     fmt.Println()
54 }
55

```

```
PS C:\Users\ACER> go run "e:\golang semester 3\modul15_5.go"
Masukkan perolehan suara untuk setiap partai (akhiri dengan -1):
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 -1

Hasil perolehan suara:
(8:15)
PS C:\Users\ACER> 
```